Journal of Machine Learning Research 14 (2013) 2355-2359 Submitted 2/12; Revised 5/13; Published 8/13

Tapkee: An Efficient Dimension Reduction Library

Sergey Lisitsyn LISITSYN.S.O@GMAIL.COM
Samara State Aerospace University
34 Moskovskoye shosse
443086 Samara, Russia

Christian Widmer* CWIDMER@CBIO.MSKCC.ORG
Computational Biology Center
Memorial Sloan-Kettering Cancer Center
415 E 68th street
New York City 10065 NY, USA

Fernando J. Iglesias Garcia FJIG@KTH.SE
KTH Royal Institute of Technology
79 Valhallavägen
10044 Stockholm, Sweden

Editor: Cheng Soon Ong

Abstract

We present *Tapkee*, a C++ template library that provides efficient implementations of more than 20 widely used dimensionality reduction techniques ranging from *Locally Linear Embedding* (Roweis and Saul, 2000) and *Isomap* (de Silva and Tenenbaum, 2002) to the recently introduced *Barnes-Hut-SNE* (van der Maaten, 2013). Our library was designed with a focus on performance and flexibility. For performance, we combine efficient multi-core algorithms, modern data structures and state-of-the-art low-level libraries. To achieve flexibility, we designed a clean interface for applying methods to user data and provide a callback API that facilitates integration with the library. The library is freely available as open-source software and is distributed under the permissive *BSD 3-clause* license. We encourage the integration of *Tapkee* into other open-source toolboxes and libraries. For example, *Tapkee* has been integrated into the codebase of the *Shogun* toolbox (Sonnenburg et al., 2010), giving us access to a rich set of kernels, distance measures and bindings to common programming languages including Python, Octave, Matlab, R, Java, C#, Ruby, Perl and Lua. Source code, examples and documentation are available at `http://tapkee.lisitsyn.me`.

Keywords: dimensionality reduction, machine learning, C++, open source software

1. Introduction

The aim of dimension reduction is to find low-dimensional representations of data to facilitate data visualization, interpretation and preprocessing for further analysis. It has applications in diverse fields including bioinformatics, physics and computer vision. Our library, *Tapkee*, provides numerous efficient implementations of both linear and non-linear dimension reduction algorithms, ranging from established methods to more recent developments in the field.

Implementations of modern dimensionality reduction algorithms are available in several other open-source toolboxes, such as *Scikit-learn* (Pedregosa et al., 2011), *Waffles* (Gashler, 2011) and the

∗. Also with Machine Learning Group, Technische Universität Berlin, Franklinstr. 28/29, 10587 Berlin, Germany.

Matlab Toolbox for Dimensionality Reduction by Laurens van der Maaten. In contrast to existing toolkits, our aim is to provide a generic C++ library in the spirit of the *Standard Template Library* to allow for greater flexibility. In particular, we designed *Tapkee* to be callback-centric (i.e., externally defined functions may be passed to the library as arguments), which naturally enables the user to combine our library with custom distance or similarity measures, which are at the core of the most dimensionality reduction algorithms. Our second central design goal is efficiency. For this, we combine advanced data structures (such as cover tree by Beygelzimer et al., 2006), with efficient libraries (as ARPACK) and carefully engineered, well-tested C++ code. Furthermore, we provide parallel implementations of many algorithms using *OpenMP* and preview preliminary support for *GPU* computing. Finally, we want *Tapkee* to be easy to use. Therefore, we provide a simple, well-documented API and usage examples for each method. To support the user in the choice of algorithms, we provide mathematical background for each method on the project website.

To facilitate the use of *Tapkee* for the end-user, we distribute our library as part of the *Shogun* toolbox, providing access to its language bindings to such languages as Python, MATLAB, Java and a rich set of kernel and distance functions. However, we emphasize that *Tapkee* is a stand-alone library. The integration of *Tapkee* into C++ projects or other toolkits is therefore highly encouraged by its software design (a template library with callbacks and only few dependencies) as well as the choice of a permissive software license (*BSD*). Source code, documentation and various graphical and code examples can be found on the project website: `http://tapkee.lisitsyn.me`.

2. Implemented Algorithms

Currently, our toolkit provides implementations of the following algorithms:

1. Local neighborhood methods, such as *Locally Linear Embedding* (Roweis and Saul, 2000) and *Neighborhood Preserving Embedding* (He et al., 2005). This group also includes *Hessian Locally Linear Embedding* (Donoho and Grimes, 2003), *Local Tangent Space Alignment* (Zhang and Zha, 2004) and *Linear Local Tangent Space Alignment* (Zhang et al., 2007);

2. Classic distance-based methods such as *Multidimensional Scaling* and *Isomap* along with its landmark approximations such as *Landmark Multidimensional Scaling* and *Landmark Isomap* (de Silva and Tenenbaum, 2002);

3. Graph-based methods, such as *Laplacian Eigenmaps* (Belkin and Niyogi, 2002), *Locality Preserving Projections* (He and Niyogi, 2003) and *Diffusion Maps* (Coifman and Lafon, 2006);

4. Iterative methods such as *Stochastic Proximity Embedding* (Agrafiotis, 2003), *Factor Analysis*, *t-SNE* and *Barnes-Hut-SNE* (van der Maaten, 2013);

5. Widely known *PCA* and *kernel PCA*, *Randomized PCA* (Halko et al., 2011) algorithms.

3. Software Design

Tapkee is a C++ pure template library with a flexible, modular structure. In practice, this allows easier integration of the library code into existing projects without the need for linking. Furthermore, the library code is specialized during compilation with capabilities of compile-time optimization and customization (e.g., custom floating point precision or linear algebra solvers). As a major design

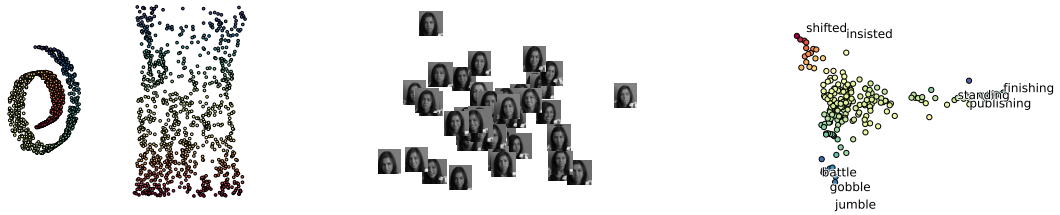

Figure 1: Swissroll 3D data and its embedding constructed with *Isomap* (on the left); embedding of the faces data set computed with *t-SNE* (in the middle); embedding of English words computed with *kernel LLE* using *Ratcliff-Obershelp* as similarity measure (on the right).

principle, our library is engineered to be *callback-based* (with externally defined functions passed to the library as arguments). The algorithms in our library are implemented as generically as possible (formulated in terms of callback functions) with callbacks providing the interface between the user data and algorithm. For example, the user may want to embed biological sequences using a string kernel from a third party library, which—using callbacks—can be passed to *Tapkee* as a custom similarity measure. A major benefit of our callback-centric design is the improved re-usability of our software, as user-defined callback functions may contain complex custom code or operate on custom data structures without any changes to our library code.

Most dimensionality reduction algorithms rely on linear algebra operations, nearest neighbor finding and the solving of eigenproblems. To address the need for high performance linear algebra we leverage the *Eigen3* template library. This makes our implementations safe, easy to read and fast. Next, we approach the nearest neighbor problem with the *vantage point tree* and the *cover tree* data structures (Beygelzimer et al., 2006) which can be used with arbitrary metric spaces. Finally, to solve eigenproblems arising in most of the implemented algorithms we employ three methods: QR decomposition, Lanczos method from the *ARPACK* library and a randomized method described in Halko et al. (2011). The library supports all major platforms (*Linux*, *Mac OS X* and *Windows*). To ensure quality we combine unit-testing and continuous integration with *Travis*.

4. Applications

We have successfully applied *Tapkee* to compute low-dimensional representations of various data sets including images and biological sequences. Images were taken from the widely-known *MNIST* data set. To demonstrate an embedding of string objects, we computed a 2*D* embedding of English words and biological sequences (gene starts) for several organisms. See Figure 1 for examples of 2*D* representations computed with *Tapkee* and the website for more graphical examples. At the time of writing, we are aware of several successful applications of *Tapkee* to problems in hydrodynamics, bioinformatics and exploratory analysis of energy landscapes, from user feedback.

5. Comparison To Related Work

We have compared our implementations with their counterparts in *Scikit-learn* 0.13.1, *Matlab toolbox for Dimensionality Reduction* 0.8 and *Waffles* 2013-04-06. We present a performance com-

	Tapkee		*Scikit-learn*		*Waffles*		*MTfDR*	
	LLE	ISOMAP	LLE	ISOMAP	LLE	ISOMAP	LLE	ISOMAP
Swissroll, $k = 15$	**0.45** (0.68)	**3.84** (14.60)	0.89	20.71	–	227.71	11.88	2433.31
MIT-CBCL, $k = 20$	**1.15** (2.57)	**1.20** (3.06)	12.77	5.94	64.76	20.05	2.75	5.56
MNIST, $k = 20$	**2.55** (3.63)	**1.93** (4.67)	8.40	9.12	–	29.92	2.80	253.55
AVIRIS, $k = 80$	**5.37** (6.27)	**7.44** (19.85)	10.76	21.53	–	62.25	25.42	763.55

Table 1: Performance comparison for *Tapkee*, *Scikit-learn* (Pedregosa et al., 2011), *Waffles* (Gashler, 2011) and *Matlab Toolbox for Dimensionality Reduction* (*MTfDR*). Time is given in seconds (− means no convergence in 3 hours). For *Tapkee*, results for 4 threads and for a single thread (in parentheses) are shown.

parison[1] for implementations of two widely-used methods, which were present in all of the above libraries: Locally Linear Embedding and Isomap. For this comparison we used four data sets: a subset of *MNIST* (2000 vectors of size 784), swissroll with 5000 3D vectors, images from *MIT-CBCL* face recognition data set (384 vectors of size 40,000) and subsampled hyperspectral image obtained from the *AVIRIS* project (2520 vectors of size 224). We report convergence times for each implementation, method and data set in Table 1. We note that when using a single thread *Tapkee* outperforms the other implementations in all but one case. When using 4 cores, *Tapkee* is considerably faster and outperforms the other libraries in all experiments. *Scikit-learn* has the second best overall performance, which hints at the high quality of code. Furthermore, *MTfDR* has good performance for high-dimensional data sets, but shows slow convergence for problems with a large number of vectors. Finally, *Waffles*' LLE implementation sometimes fails to converge - this may be caused by the use of the power iteration method.

In summary, we find that the choice of algorithms and low-level libraries enables *Tapkee* to outperform other implementations under various conditions. For the code to reproduce the above experiments (and additional benchmarks) and an overview of available methods for each toolkit, please see our supplementary webpage `http://iglesias.github.io/tapkee_benchmarks/`.

6. Conclusion

We have implemented an efficient and flexible library for dimension reduction with all the methods implemented using state-of-the-art algorithms and data structures. Our library readily handles big data sets and facilitates interaction with custom code using a callback-centric API. To compare our toolkit to existing software, we provide a speed comparison on a range of data sets showing considerable speed-up. We believe the infrastructure provided by *Tapkee* (such as API, modular design, bindings to libraries) can serve as a platform for further dimension reduction development. Finally, we hope that our work proves to be valuable to researchers, students and practitioners.

Acknowledgments

We would like to thank: L. van der Maaten, J. Langford, V. Gorbatiuk, E. Andreev, D. Harmon and D. Okanohara for their code; G. Rätsch, K.-R. Müller, S. Sonnenburg and D. Kuo for helpful comments. The work was supported by Google (GSoC 2011), the Russian Foundation for Basic Research (13-07-12030, 12-07-00581-a), the Max Planck Society and DFG grant RA1894/1.

1. We used a machine with 2 Intel® Xeon® CPUs and 16Gb of RAM running Ubuntu 12.04.2 and Matlab® 2011a.

References

D. K. Agrafiotis. Stochastic proximity embedding. *Journal of Computational Chemistry*, 24(10): 1215–1221, 2003.

M. Belkin and P. Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. *Science*, 14:585–591, 2002. ISSN 10495258.

A. Beygelzimer, S. Kakade, and J. Langford. Cover trees for nearest neighbor. *Proceedings of the 23rd International Conference on Machine Learning ICML 06*, 1:97–104, 2006.

R. Coifman and S. Lafon. Diffusion maps. *Applied and Computational Harmonic Analysis*, 21(1): 5–30, 2006.

V. de Silva and J. B. Tenenbaum. Global versus local methods in nonlinear dimensionality reduction. *Advances in Neural Information Processing Systems*, 15:705–712, 2002.

D. L. Donoho and C. Grimes. Hessian eigenmaps: Locally linear embedding techniques for high-dimensional data. *Proceedings of the National Academy of Sciences of the United States of America*, 100(10):5591–5596, 2003.

M. S. Gashler. Waffles: A machine learning toolkit. *Journal of Machine Learning Research*, 12 (July):2383–2387, 2011.

N. Halko, P.-G. Martinsson, Y. Shkolnisky, and M. Tygert. An algorithm for the principal component analysis of large data sets. *SIAM Journal on Scientific Computing*, 33(5):2580–2594, 2011.

X. He and P. Niyogi. Locality preserving projections. *Matrix*, 16(December):153–160, 2003.

X. He, D. Cai, S. Yan, and H.-J. Zhang. Neighborhood preserving embedding. *Tenth IEEE International Conference on Computer Vision ICCV05 Volume 1*, 2:1208–1213, 2005.

F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

S. T. Roweis and L. K. Saul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

S. Sonnenburg, G. Raetsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. De Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 11(x):1799–1802, 2010.

L. van der Maaten. Barnes-Hut-SNE. *arXiv preprint arXiv:1301.3342*, 2013.

T. Zhang, J. Yang, D. Zhao, and X. Ge. Linear local tangent space alignment and application to face recognition. *Neurocomputing*, 70(7-9):1547–1553, 2007. ISSN 09252312.

Z. Zhang and H. Zha. Principal manifolds and nonlinear dimensionality reduction via tangent space alignment. *SIAM Journal on Scientific Computing*, 26(1):313–338, 2004.