DynoGraph: Dynamic Graph Construction for Nonlinear Dimensionality Reduction - Supplementary

Li Qian*†, Claudia Plant‡, Yalan Qin§, Jing Qian¶, Christian Böhm†

*Institute of Informatics, Ludwig Maximilian University of Munich, Munich, Germany, li.qian@dbs.ifi.lmu.de
†Faculty of Computer Science, University of Vienna, Vienna, Austria, christian.boehm@univie.ac.at
‡Faculty of Computer Science, ds:Univie, University of Vienna, Vienna, Austria, claudia.plant@univie.ac.at
§School of Communication and Information Engineering, Shanghai University, Shanghai, China, ylqin@shu.edu.cn
¶Carbon Neutral R&D Center, China Railway Hi-Tech Industry Corporation Limited, Beijing, China, jing.qian@crhic.cn

I. Implementation and Experimental Setup

A. Experimental Environment

The DynoGraph algorithm is implemented in Python, utilizing the `faiss-gpu` library [1] for nearest neighbor search. All experiments were conducted on a machine equipped with a 14-core Intel Core i9 2.50 GHz CPU, 64GB of RAM, and an NVIDIA GeForce RTX 3080 Ti GPU. For visualization and comprehensive comparison, all experiments aim to learn a two-dimensional embedding representation of the original data.

B. Datasets

We select six real-world datasets from various domains to demonstrate the broad applicability of DynoGraph. The WarpPIE10P [2] dataset is used for facial recognition. From the UCI Machine Learning Repository [3], we include LandsatSatellite and Human Activity Recognition Using Smartphones (HAR) datasets, which represent satellite images and activity recognition using time series data collected from smartphone sensors, respectively. The COIL-20 dataset [4], developed by Columbia University, consists of 20 objects captured from different angles, commonly used for object recognition tasks. Additionally, we use Fashion-MNIST [5] and MNIST [6] datasets for image classification of clothing and handwritten digits, respectively. The features range from 36 to 16384, and the statistics of the datasets are presented in Table I. For image data, we scale the pixel values to the range $[0, 1]$ by dividing by the maximum pixel value. For multivariate data, we standardize each feature by removing the mean and scaling to unit variance [7].

TABLE I: Characteristics of datasets.

Dataset	# Instances	# Features	# Classes
3D Scurve_hole	8540	3	1
WarpPIE10P	210	2420	10
COIL-20	1440	16384	20
LandsatSatellite	6435	36	6
HAR	10299	561	6
Fashion-MNIST	70000	784	10
MNIST	70000	784	10

C. Baselines

To evaluate the effectiveness of DynoGraph, we conduct a comprehensive comparative analysis against nine classical or state-of-the-art dimensionality reduction techniques. These include PCA [8][1], MDS [9][1], LLE [10][1], Eigenmaps [11][1], t-SNE [12][1], LargeVis [13][2], UMAP [14][3], TriMap [15][4], and SpaceMAP [16][5]. To ensure consistency and fairness in comparisons, we parameterize all comparison methods as recommended in the corresponding publications. DynoGraph does not require any input parameters.

D. Evaluation Metric

We evaluate the effectiveness of dimensionality reduction algorithms using two global evaluation metrics, including Procrustes analysis [17] and Adjusted Mutual Information (AMI) [18], and a local evaluation metric, k-nearest neighbors (kNN) classifier accuracy [19].

For the synthetic dataset, we generate two-dimensional ground truth coordinates and fold them to three-dimensional space through a nonlinear transformation. We use Procrustes analysis to compare the difference between the ground truth coordinates and those obtained by the dimensionality reduction algorithms. The Procrustes analysis reports the mean square error after optimally aligning the embedding to the ground truth using translation, rotation, reflection and scaling, with error values ranging from 0 to 1. Lower Procrustes errors indicate greater faithfulness to the global structure of the original data.

For real datasets, we perform k-means clustering (initialized with k-means++) and use AMI to evaluate the clustering

[1] https://scikit-learn.org/stable/

[2] https://github.com/lferry007/LargeVis

[3] https://github.com/lmcinnes/umap

[4] https://github.com/eamid/trimap

[5] https://github.com/zuxinrui/SpaceMAP

performance of the low-dimensional embedding. AMI values range from 0 to 1, with higher values indicating higher consistency between the clustering results and the true labels, highlighting the effectiveness of the dimensionality reduction algorithms in preserving global structure. Additionally, we use the 10-fold cross-validated kNN classifier accuracy to evaluate the performance in terms of preserving local structure. The choice of k depends on the number of data points n in the dataset, ranging from 1% to 20%. The range of the k-NN classifier accuracy is from 0 to 1, with higher accuracies indicating better performance in preserving local structures. All experiments were performed ten times and we report the average results.

References

[1] J. Johnson, M. Douze, and H. Jégou, “Billion-scale similarity search with GPUs,” *IEEE TBD*, vol. 7, no. 3, pp. 535–547, 2019.

[2] T. Sim, S. Baker, and M. Bsat, “The CMU pose, illumination, and expression (PIE) database,” in *FGR*, 2002, pp. 53–58.

[3] D. Dua and C. Graff, “UCI machine learning repository,” 2017.

[4] S. A. Nene, S. K. Nayar, and H. Murase, “Columbia object image library (coil-20),” 1996.

[5] H. Xiao, K. Rasul, and R. Vollgraf, “Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms,” *ArXiv Preprint ArXiv:1708.07747*, 2017.

[6] Y. LeCun, C. Cortes, and C. J. C. Burges, “The mnist database of handwritten digits,” http://yann.lecun.com/exdb/mnist/, 1998.

[7] S. García, J. Luengo, and F. Herrera, *Data preprocessing in data mining*. Springer, 2015.

[8] L. I. Smith, “A tutorial on principal components analysis,” 2002.

[9] I. Borg and P. J. Groenen, *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[10] S. T. Roweis and L. K. Saul, “Nonlinear dimensionality reduction by locally linear embedding,” *Science*, vol. 290, no. 5500, pp. 2323–2326, 2000.

[11] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, 2003.

[12] L. van der Maaten and G. Hinton, “Visualizing data using t-sne,” *JMLR*, vol. 9, no. 86, pp. 2579–2605, 2008.

[13] J. Tang, J. Liu, M. Zhang, and Q. Mei, “Visualizing large-scale and high-dimensional data,” in *WWW*, 2016, pp. 287–297.

[14] L. McInnes, J. Healy, and J. Melville, “UMAP: uniform manifold approximation and projection for dimension reduction,” *ArXiv Preprint ArXiv:1802.03426*, 2018.

[15] E. Amid and M. K. Warmuth, “Trimap: Large-scale dimensionality reduction using triplets,” *ArXiv Preprint ArXiv:1910.00204*, 2019.

[16] X. Zu and Q. Tao, “SpaceMAP: Visualizing high-dimensional data by space expansion,” in *ICML*, 2022, pp. 27 707–27 723.

[17] F. Bai and A. Bartoli, “Procrustes analysis with deformations: A closed-form solution by eigenvalue decomposition,” *IJCV*, vol. 130, no. 2, pp. 567–593, 2022.

[18] N. X. Vinh, J. Epps, and J. Bailey, “Information theoretic measures for clusterings comparison: Variants, properties, normalization and correction for chance,” *JMLR*, vol. 11, p. 2837–2854, 2010.

[19] P. Cunningham and S. J. Delany, “k-nearest neighbour classifiers-a tutorial,” *ACM Comput. Surv.*, vol. 54, no. 6, pp. 1–25, 2021.