A Tool for Visualizing and Analyzing High-Dimensional Clustering Performance

Justin Lin*and Julia Fukuyama†

Abstract

Technological advances have spurred an increase in data complexity and dimensionality. We are now in an era in which data sets containing thousands of features are commonplace. To digest and analyze such high-dimensional data, dimension reduction techniques have been developed and advanced along with computational power. Of these techniques, nonlinear methods are most commonly employed when working with high-dimensional data because of their ability to construct visual two-dimensional embeddings of high-dimensional data. These methods unevenly stretch and shrink space in a way that represents the data's structure in a fewer number of dimensions. However, attempting to capture high-dimensional structures in a significantly lower number of dimensions requires drastic manipulation of space. As such, nonlinear dimension reduction methods are known to occasionally capture false structures, especially in noisy settings. In efforts to deal with this phenomenon, we developed an interactive tool that enables analysts to better understand and diagnose their dimension reduction results. It uses various analytical plots to provide a multi-faceted perspective on captured structures in the data to determine if they're faithful to the original data or remnants of the dimension reduction process. The tool is available in an R package named *insert name here*.

1 Introduction

The potency of nonlinear dimension reduction methods lies in their flexibility, allowing them to model complex data structures. That same flexibility, however, makes them difficult to use and interpret. Each method requires a slew of hyperparameters that need to be calibrated, and even when adequately calibrated, these methods require a trained eye to interpret. For example, the two most popular nonlinear dimension reduction methods, t-SNE and UMAP, are known to generate unintuitive results ([5], [17]). The results often cluster, even when no clusters exist in the data. Moreover, cluster sizes and inter-cluster distances can be unreliable. We've developed an interactive tool that analysts may use to conduct a post-hoc analysis of their high-dimensional clustering. The tool uses the minimum spanning tree (MST) to model the global structure of clusters and to provide an additional perspective on inter-cluster relationships. This allows analysts to extract more information from their dimension reduction results by making it easier to differentiate the signal and the noise.

2 Methods

2.1 The Minimum Spanning Tree

Graphs have been applied to many multivariate statistical problems. The authors of [15] introduced the minimal ascending path spanning tree as a way to test for multimodality. The Friedman-Rafsky test [9], along with its modern variations [2, 3, 4], use the MST to construct a multivariate two-sample test. Single-linkage clustering [8] and runt pruning [12] are both intimately related with the MST. In the context of dimension reduction, IsoMap [13] makes use of neighborhood graphs, [10] introduces the maximum information spanning tree, and [14] uses the MST. These methods, which fall under the category of manifold learning, use graphs to model high-dimensional data assumed to be drawn uniformly from a non-linear manifold. An accurate low-dimensional embedding can then be constructed from these

*Department of Mathematics, Indiana University
†Department of Statistics, Indiana University

graphs. It's apparent that graphs are useful for modeling high-dimensional data, especially when it comes to dimension reduction and cluster analysis. Our tool uses the MST to analyze the reliability of visualizations produced by nonlinear dimension reduction methods.

To construct the minimum spanning tree (MST), start with a set of vertices, one for each point in the data set. When an edge is added between two vertices, it's assigned a weight equal to the dissimilarity between the corresponding points. The edges of the MST are selected from all possible edges so that the sum of edge weights is minimized and there exists a path between any two vertices. We've opted for the MST for a couple of key properties. Firstly, the MST and shortest paths along it are quick to compute. Secondly, the MST contains a unique path between any two vertices, providing a well-defined metric on the data. Lastly, it provides a good summary of the data's structure. It contains as a subgraph the nearest-neighbor graph, and any edge deletion in the MST partitions the vertices into two sets for which the deleted edge is the shortest distance between them [9].

2.1.1 MST Stability

The MST is meant to provide a robust estimation of the data's global structure, and more specifically, inter-cluster relationships. As such, it should be stable in the presence of noise and unaffected by local transformations of the data. To demonstrate MST stability, we study the effect of random noise on the inter-cluster relationships explained by the MST.

To derive the inter-cluster relationships from the MST, we simplified the medoid subtree using the following procedure:

Algorithm 1 Simplified Medoid Subtree

Require: MST $T = (V, E)$ with medoid vertices $m_1, \ldots, m_k \in V$

$T' = (V', E') \Leftarrow$ minimal subtree of T containing all medoid vertices m_i

perplexities $\Leftarrow \{p_1, \ldots, p_m\}$

repeat

  Let $v \in V'$ with $deg(v) = 2$ and neighbors $a, b \in V'$. Let $d(v, a)$ and $d(v, b)$ be the weights of the edges connected v and to a and b.

  Replace v and its two incident edges with an edge connecting a and b with weight $d(v, a)$ and $d(v, b)$.

until T' contains no longer contains non-medoid vertices with degree two

output T'

The simplification process essentially replaces paths of non-medoid vertices with one edge of equal length. We refer to this tree as the simplified medoid subtree and is meant to encode global inter-cluster relationships.

2.1.2 Robinson-Foulds Metric

To compare simplified medoid subtrees, we used the Robinson-Foulds metric [16]. The R-F metric was originally introduced to quantify the dissimilarity of phylogenetic trees, but the algorithm generalizes to arbitrary weighted trees. It looks at partitions of each tree created by removing individual edges, then counts the number of partitions present in one tree but not in the other. We modified the algorithm (Algorithm 2) to specifically measure the dissimilarity in medoid vertices and applied a normalization so that the distances range from zero to one.

Algorithm 2 Robinson-Foulds Distance

Require: Trees $T_1 = (V_1, E_1)$ and $T_2 = (V_2, E_2)$ with medoids $m_1, \ldots, m_k \in V_1$ and $n_1, \ldots, n_k \in V_2$

$P_1 \Leftarrow \{\}$
for $e \in E_1$ **do**
  $G \Leftarrow (V_1, E_1 \setminus \{e\})$ with connected components G_1 and G_2
  $M_1 \Leftarrow \{m_1, \ldots, m_k\} \cap V(G_1)$
  $M_2 \Leftarrow \{m_1, \ldots, m_k\} \cap V(G_2)$
  $P_1 \Leftarrow \text{ADD}(P_1, \{M_1, M_2\})$
$P_2 \Leftarrow \{\}$
for $e \in E_2$ **do**
  $G \Leftarrow (V_2, E_2 \setminus \{e\})$ with connected components G_1 and G_2
  $M_1 \Leftarrow \{n_1, \ldots, n_k\} \cap V(G_1)$
  $M_2 \Leftarrow \{n_1, \ldots, n_k\} \cap V(G_2)$
  $P_2 \Leftarrow \text{ADD}(P_2, \{M_1, M_2\})$
output $\frac{|P_1 \Delta P_2|}{2|P_1 \cap P_2|}$

2.1.3 MST Stability Experiment

1,500 samples were randomly chosen from the MNIST data set [6]. Each 784×784-pixel image was flattened into a vector of length 784^2, so the data contain 1,500 samples in 784^2 dimensions. A PCA pre-processing step was employed to reduce the number of dimensions to 300. The simplified medoid subtree T was then calculated.

Random Gaussian noise was then added to the data and the new simplified medoid subtree T' was calculated. The R-F distance $RF(T, T')$ was recorded. This process was repeated 30 times.

To better interpret the R-F distances, we designed a null distribution of distances as a reference for comparison. These distances should represent R-F distances between trees that do not portray similar global structures and inter-cluster relationships. To generate the null distribution from the data, we randomly permuted the class labels and computed the R-F distances between the resulting simplified medoid subtrees and the original simplified medoid subtree. By randomly re-labelling the clusters, we are simulating examples with distinct global structures. Figure 1 shows the R-F distances produced by adding noise and permuting the class labels. The simplified medoid subtree trees generated by adding noise were significantly closer to the original simplified medoid subtree than those generated by randomly permuting the class labels in terms of R-F distance.

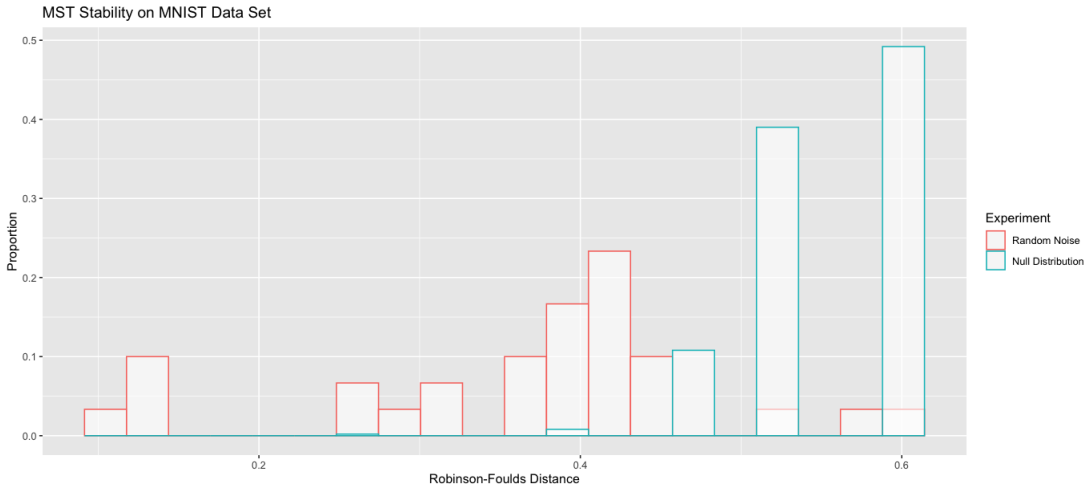


Figure 1: MST Stability on MNIST Data Set

2.2 The Inputs

The interactive tool takes as input a dissimilarity matrix $D \in \mathbb{R}^{n \times n}$ representing the dissimilarities between the n high-dimensional points, a two-dimensional embedding $X \in \mathbb{R}^{n \times 2}$, and a clustering $\mathcal{C} \in \{1, \ldots, k\}^n$ where k is the number of classes. Optionally, it also takes a set of ID's to denote the points.

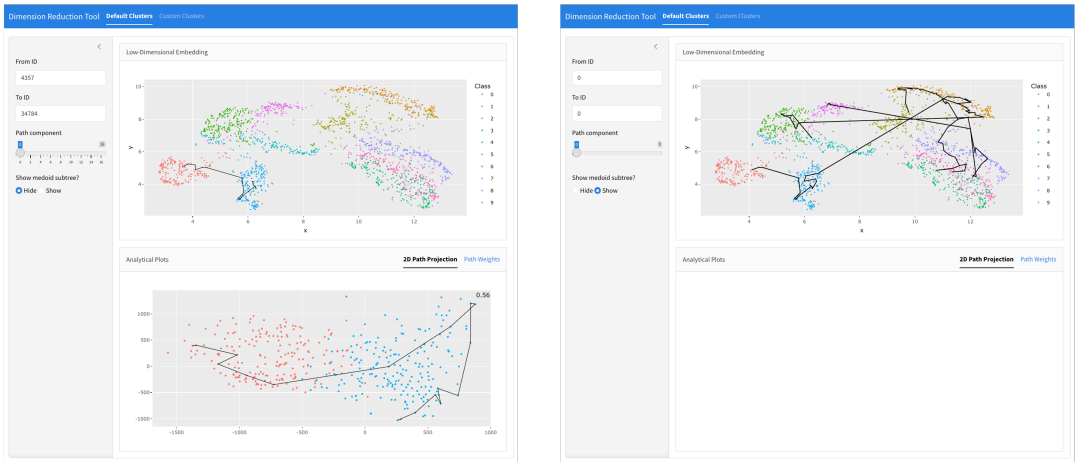
The dissimilarity matrix must be a symmetric matrix with 0's along the diagonal. The entry $D_{ij} = D_{ji}$ should represent the dissimilarity or distance between points i and j. The chosen dissimilarity measure should be appropriate for the type of data. For example, the L_1 norm and even fractional L_p distance measures have been suggested for high-dimensional numerical data [1], Jaccard and cosine metrics are popular when working with text data [11], and a variety of Image Distance Functions (IDFs) have been suggested for image data [7]. From this dissimilarity matrix, the MST is calculated.

2.3 The Dashboard

The dashboard contains two main panels as well as a side panel including adjustable settings. The first main panel depicts the low-dimensional embedding colored according to the provided clustering. The second main panel contains analytical plots. When supplied with a pair of points via "From ID" and "To ID", the MST path is calculated between the high-dimensional points and corresponding path in low-dimension is drawn upon the low-dimensional embedding (Figure 2a). The first analytical plot depicts a PCA projection of the high-dimensional path along with the clusters the two endpoints belong to. The PCA transform is calculated using only the points along the path then applied to the rest of the points. The number in the top right corner is the proportion of variance retained by the PCA projection. The second analytical plot contains a bar plot of path weights. The user may cycle through the segments along the path using the "Path component" slider in the side panel. The corresponding segment will be highlighted in both the low-dimensional embedding and the 2D path projection. The corresponding path weight will also be highlighted in the plot of path weights.

To get a holistic view of the data's global structure, the user may view the medoid subtree by selecting "Show" in the side panel (Figure 2b). The medoid subtree is the minimal subtree of the MST containing the medoid of each class. The medoid subtree describes the global arrangement of clusters.

Along the top bar, the user may also navigate to the next page named "Custom Clusters". This page contains the same plots, but the user may instead define their own clusters of interest. To define a cluster, the user must select a subset of points by clicking and dragging along the low-dimensional embedding plot. Once the points are selected, the user can save the selected points by clicking "Submit Group 1". The user must then repeat the process with the second cluster of points and click "Submit Group 2". Once both clusters have been submitted, the path between the (high-dimensional) medoids of the selected clusters is portrayed. The user may also adjust the endpoints of the path. The 2D path projection will contain the points along the path together with the selected points. This page is highly useful for analyzing spacial clusters that may not be represented by the provided clustering.



Figures 2a and 2b: Dashboard

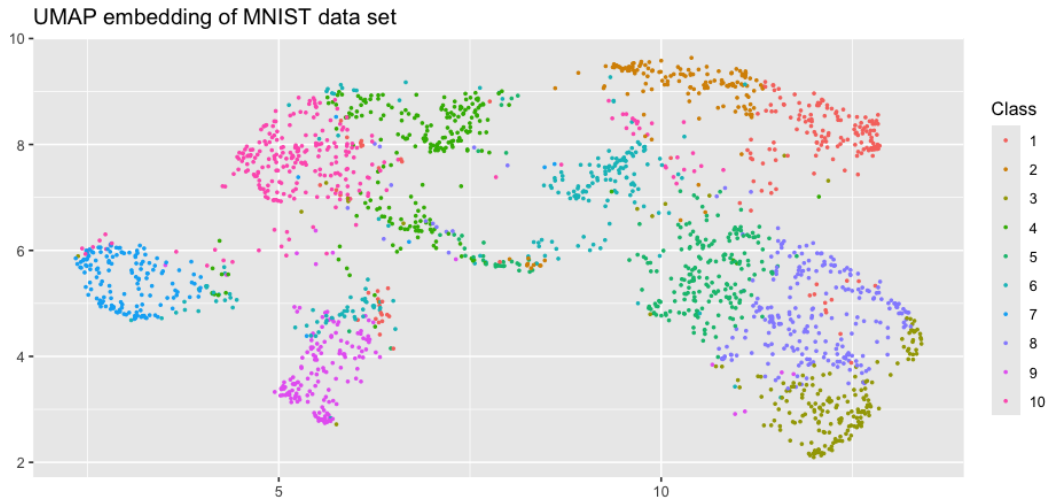


Figure 3: k-Means Applied to MNIST Data

3 Results

3.1 MNIST

The MNIST database is a collection of 70,000 images of handwritten digits [6]. Each image contains 784×784 pixels, so when vectorized, the data set contains 70,000 points in 614,656 dimensions. A random subsample of 2,000 images was taken and PCA pre-processing step was applied. The number of dimensions was reduced to 300 before applying UMAP to construct a two-dimensional visualization.

To demonstrate how to use the tool, we analyze a hypothetical clustering instead of the true class labels. k-means was applied to the processed high-dimensional data with 10 clusters, one for each digit. k-means does not agree with the UMAP embedding for certain clusters (Figure 3).

3.1.1 Classes One and Two

The UMAP embedding suggests classes one and two could be combined into one cluster. k-Means is known to struggle with clusters of varying sizes and densities, so it may have incorrectly separated this cluster into two classes.

To start, we navigate to the custom clusters tab and select the class one and class six points as our two groups. The path connecting the high-dimensional medoids is projected on the low-dimensional plot. The path exhibits minimal complexity and does not pass through any other clusters. Moreover, the path weights are consistent and relatively small (the path weights are scaled relative to the longest path in the MST). The edge connecting the two classes is also no longer than other edges along the path. This implies the class one and class two points are tightly packed, suggesting they represent the same digit.

3.1.2 Class Four

The class four points are split into two separate clusters. To understand their relationship, we select the clusters as our two groups then select endpoints contained in the center of each cluster. The resulting path is long and complex while also passing through various other clusters. Though the complexity is exaggerated by the low-dimensional embedding. Inspection of the path weights reveals the other clusters the path passes through are not as far away from the class four clusters as depicted. In fact, the longest segments along the path are contained within the two class four clusters. k-means is known to struggle with data with nonuniform density, so these points were grouped together due to their low density. While the circuitous path suggests the class four clusters represent separate digits, the clusters may still represent similarly looking digits.

3.1.3 Classes Three, Five, and Eight

Looking at classes three, five, and eight, the upper portions of each class seem to form an elongated cluster together. To determine if this is a cohesive cluster, we select half of the cluster as group one and the other half as group two. We also switch the endpoints to construct a path that spans the entirety of the cluster while also ensuring all three classes are represented in the 2D path projection plot.

The resulting path passes from class five, to class eight, then to class three. Along the way, the path also passes through a couple class one points, some within the elongated cluster and some outside. Investigation of the path weights reveals that the exterior class one points aren't nearly as far removed as depicted. The low-dimensional separation is a result of the non-linear dimension reduction. Therefore, the path connecting the endpoints of the elongated cluster is relatively direct with minimal complexity, suggesting the upper portions of classes three, five, and eight represent the same digit.

The natural follow-up is to investigate the lower portions of classes three, five, and eight. Again, we split the cluster in half to serve as our two groups and adjust the endpoints to get a path that spans the entire cluster. The resulting path passes through classes one and two as well as the cluster consisting of the upper portions of classes three, five, and eight. The path weights, as discussed previously, show the low-dimensional embedding exaggerates the distance between classes one/two and classes three/five/eight. While the path passes through other clusters, they are all nearby, so there is a lack of conclusive evidence.

3.2 The True Class Labels

To verify the analysis, we reference the true class labels. It was correct to combine classes one and six. Together, they represent the digit one. It was correct to separate class four into two separate clusters. The two clusters represent digits five and eight, which aligns with our hypothesis that the clusters are separate, but represent similarly looking digits. The upper portions of classes three, five, and eight do, in fact, represent the same digit. They represent the digit seven. And lastly, the lower portions of classes three, five, and eight represent the digits three and nine. These digits were tough to distinguish and very similar numerically, as illustrated by their overlapping nature in the low-dimensional embedding.

References

[1] Charu C. Aggarwal, Alexander Hinneburg, and Daniel A. Keim. On the surprising behavior of distance metrics in high dimensional space. *Lecture Notes in Computer Science, vol. 1973*, 2001.

[2] Bhaswar B. Bhattacharya. A general asymptotic framework for distribution-free graph-based two-sample tests. *Journal of the Royal Statistical Society Series B: Statistical Methodology 81:3, 575-602*, 2019.

[3] Hao Chen and Jerome H. Friedman. A new graph-based two-sample test for multivariate and object data. *Journal of the American Statistical Association 112:517, 397-409*, 2017.

[4] Hao Chen, Xu Chen, and Yi Su. A weighted edge-count two-sample test for multivariate and object data. *Journal of the American Statistical Association 113:523, 1146-1155*, 2018.

[5] Andy Coenen and Adam Pearce for Google PAIR. Understanding UMAP. *https://pair-code.github.io/understanding-umap/.*

[6] Li Deng. The MNIST database of handwritten digit images for machine learning research. *IEEE Signal Processing Magazine 29:6*, 2012.

[7] Vito Di Gesù and Valery Starovoitov. Distance-based functions for image comparison. *Pattern Recognition Letters 20:2, 207-214*, 1999

[8] J. C. Gower and G. J. S. Ross. Minimum spanning trees and single linkage cluster analysis. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 18:1, 54-64*, 1969.

[9] Jerome H. Friedman and Lawrence C. Rafsky. Multivariate generalizations of the Wald-Wolfowitz and Smirnov two-sample tests. *Annals of Statistics 7:4, 697-717*, 1979.

[10] Bracken M. King and Bruce Tidor. MIST: Maximum information spanning trees for dimension reduction of biological data sets. *Bioinformatics 25:9, 1165-1172*, 2009.

[11] Abdul Wahab Qurashi, Violeta Holmes, and Anju P. Johnson. Document processing: Methods for semantic text similarity analysis. *IEEE*, 2020.

[12] Werner Stuetzle. Estimating the cluster tree of a density by analyzing the minimal spanning tree of a sample. *Journal of Classification 20, 25-47*, 2003.

[13] Joshua B. Tenenbaum, Vin de Silva, and John C. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science 290:2319*, 2000.

[14] Daniel Probst and Jean-Louis Reymond. Visualization of very large high-dimensional data sets as minimum spanning trees. *Journal of Cheminformatics 12:12*, 2020.

[15] Gregory Paul M. Rozál and J.A. Hartigan. The MAP test for multimodality. *Journal of Classification 11, 5-36*, 1994.

[16] D. F. Robinson and L. R. Foulds. Comparison of Phylogenetic Trees. *Mathematical Biosciences 53, 131-147*, 1981.

[17] Martin Wattenberg, Fernanda Viégas, and Ian Johnson. How to Use t-SNE Effectively. *Distill*, 2016.