Find Distance Function, Hide Model Inference

Jingjing Liu [*]
Tufts University

Eli T. Brown[†]
Tufts University

Remco Chang[‡]
Tufts University

Abstract

Faced with a large, high-dimensional dataset, many turn to data analysis approaches that they understand less well than the domain of their data. An expert's knowledge can be leveraged into many types of analysis via a domain-specific distance function, but creating such a function is not intuitive to do by hand. We have created a system that shows an initial visualization, adapts to user feedback, and produces a distance function as a result. Specifically, we present a multidimensional scaling (MDS) visualization and an iterative feedback mechanism for a user to affect the distance function that informs the visualization without having to adjust the parameters of the visualization directly. An encouraging experimental result suggests that using this tool, data attributes with useless data are given low importance in the distance function.

1 Introduction

There are many powerful data visualization and analysis techniques at the fingertips of anyone with data to understand. Many techniques rely on a distance function. That is, these algorithms require a function that assigns a numeric distance to any two points in the input data space. To build one requires more domain expertise than an analysis specialist has, and more analysis expertise than the domain expert has.

In this work, we present a platform that allows a user not only to explore data visually, but to provide feedback that informs the underlying visualization-generating model how to adapt its distance function. The user does not have to manipulate the parameters of the model directly, but rather isolate what data points are inconsistent with her understanding of the domain, and fix them. The model gets adjusted, resulting in a new visualization for her to iteratively improve. This process allows the user to explore her data by testing hypotheses about its structure. She ultimately ends up with a useful product: a distance function which she can use for further work on her data.

There are a variety of mathematical models for visualizing high-dimensional data in two dimensions. For this poster, we chose one dimension-reduction model, multidimensional scaling (MDS) [1], which maps a high-dimensional dataset to lower-dimensions by preserving pairwise distances between datapoints accross the high- and low-dimensional spaces. The function used to compute those distances gets changed iteratively through the user's interaction with the visualization.

There are already tools that allow a user to modify the parameters of a models generating a visualization, including work by one of this paper's authors [4] [2]. The drawback of these tools is that they require the user to be an expert in the model used to generate the visualization. In this work, because we compute the effect on the distance function for the user, she does not need to know about MDS or model inference to influence the results based on her knowledge. A recent work of Endert et al. [3] created a similar visual interaction framework to ours, allowing a user to interact with a visualization to update MDS, saving the user from the agony of understanding the mathematical technicalities. Our work can be distinguished in several aspects: 1) in addition to acquiring a visualization of the dataset that fits the user's mental image, we focus on producing a distance function for the data domain that the user can use to discover hidden patterns and gain deeper understanding in further work; 2) our user-feedback adjustments are based on an objective function that not only considers the latest changes, but tries to maintain the structure of the rest of the data; 3) the interactive process can be iteratively continued until the visualization achieves the user's satisfaction, i.e., previous updates will affect final output.

2 Approach and Method

One step of the interactive process using the visual analytic tool in this work is as follows: 1) System provides a visualization based on initial values of model parameters. 2) Users observe the visualization and provide input in a predefined format. 3) System adjusts the parameters of the model to reflect the user's understanding and regenerates an updated visualization based on the new parameter values. 4) User observes this visualization and decides either to keep this modification or not.

As shown in Figure 1, the process start with receiving high-dimensional dataset as input, then iteratively updates the distance function until user is satisfied with 2D projection.

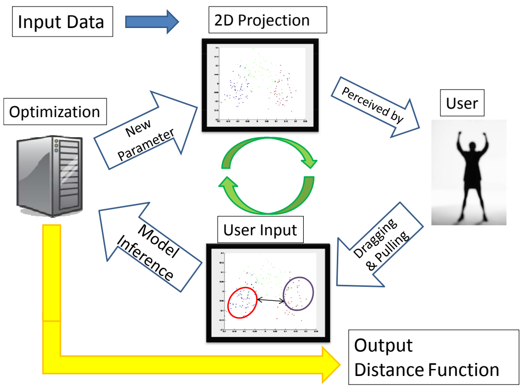

Figure 1: Flow chart showing the interactive process.

2.1 Producing the Visualization

Data visualizations for high-dimensional datasets are concerned with data $X = \{x_1, x_2, ..., x_N\}$, with each instance x_i given by an M-dimensional vector that specifies a value in each of the M features of the dataset. Alongside the data is a vector representing the relative importance of each of the features in the form of a weight vector $\Theta = [\theta_1, \theta_2, ..., \theta_M]$. A simple, linear distance function $D(x, y|\Theta)$ takes Θ as parameters and computes a real number that quantifies the dissimilarity between two data points x and y. Classic multidimensional scaling takes an input matrix giving dissimilarities between all pairs of data points and maps it to a low dimensional (in this case two-dimensional) space, minimizing a stress function [1].

[*] e-mail:jingjing.liu@tufts.edu
[†] e-mail:ebrown@cs.tufts.edu
[‡] e-mail:remco@cs.tufts.edu

The two dimensional spatial coordinates $Y = \{y_1, ..., y_N\}$ are found by solving:

$$Y = \arg\min_{Y} \sum_{i<j\leq N} \left| \|y_i - y_j\| - D(x_i, x_j|\Theta) \right| \quad (1)$$

In this work, we define our distance function as a weighted Euclidean distance:

$$D(x_i, x_j|\Theta) = \left(\sum_{m=1}^{M} \theta_m (x_{im} - x_{jm})^2 \right)^{\frac{1}{2}} \quad (2)$$

where $\sum_m \theta_m = 1$

2.2 Updating for User Adjustment

Once a visualization is generated, the user may either agree with the display and learn from aspects of the visualization as well as the weight vector Θ, or disagree with some part of the visualization based on their domain expertise. The user can interact by selecting groups of points and moving them closer together or further apart, specifically, the user input for each iteration consists two parts:

1. Two sets of data points Y_1 and Y_2 whose relative distance is not consistent with user's mental image,
2. Intention indicator $I > 0$, where I represent the ratio of *i*ntended distance between Y_1 and Y_2 and *c*urrent distance. Specifically, $I < 1$ means user wanted to move two sets closer to each other, and $I > 1$ means user wanted to move two sets further away from each other.

The assumption here is that the user wants to change the relative distance between Y_1 and Y_2 *o*nly, and he/she wants to maintain the structure for all other data points.

Once our system receives a user input, it changes the distance between Y_1 and Y_2 along the desired direction while maintaining relative distances of other points. For the t^th iteration, with user input $\{Y_1^t, Y_2^t, I^t\}$, parameter Θ is updated by solving

$$\Theta^t = \arg\min_{\Theta} \sum_{i<j\leq N} \left| D(x_i, x_j|\Theta) - I_{ij}^t \cdot D(x_i, x_j|\Theta^{t-1}) \right| \quad (3)$$

where $I_{ij}^t = I^t$ if $x_i \in Y_1$ and $x_j \in Y_2$; $I_{ij}^t = 1$ otherwise. In this work, this equation is solved by gradient descent.

3 Application and Experimental Result

Using the the system described in Section 2, we achieved feature selection for high-dimensional data by evaluating the weight vector corresponding to the visualization that satisfied the user.

The dataset used in the experiments is generated from the Wine dataset in the UCI Machine Learning Repository. This dataset contains 178 data points described by 13 features labeled into 3 classes. We added 10 random features to obtain a synthetic dataset of 23 features.

The visualization generated by weighting all features equally present overlap between pairs of classes, therefore, in this experiment, the domain expert tries to obtain a distance function that would more cleanly separate three classes. After three updates to the initial weight vector (see Figure 2(a)(b)(c)), the visualization shown in Figure 2(d) clearly separates the blue-green class pair as well as the brown-green class pair. The final step after acquiring satisfying spacial visualization is to examine the weight vector that generates this result. As shown in Figure 3, this process effectively reduced weights on random features and put high weights on features that would separate different classes, in this case the 1^{st}, 2^{nd}, 11^{th}, 12^{th} and 13^{th} feature in the original wine dataset.

4 Conclusion

In this work we introduced a system that allows a user to interact directly with a high-dimensional data visualization in order to use her expertise in the data domain to manipulate the distance function without having to understand or directly control model parameters. The product of the user's iterative manipulation is a distance function that reflects both separation in feature space and the user's domain knowledge. We performed experiments to validate the approach and found that when we introduced extraneous features to the data, they in-fact were given low priority in the resulting distance function.

Plans for further work are centered around creating a tool for discovering distance functions based on user interaction with a visualization. We intend to create a toolkit composed of several types of visual interaction that modify the distance function for the user. Further, we will examine other methods of making the projection to 2D and updating the model, based on the work of Section 2. Since recalculation can now take minutes, we will work to improve the speed dramatically. Finally, we plan to test the tool with its intended audience, people with domain expertise and high-dimensional data who do not know how to construct a distance function.

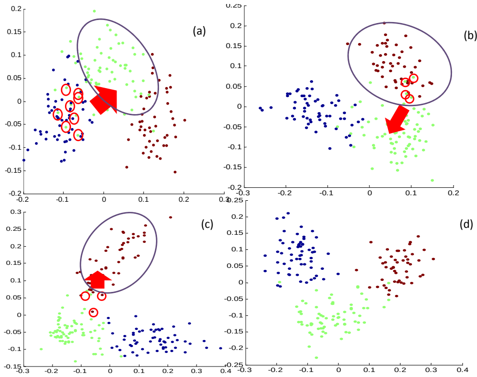

Figure 2: Interactive process to separate 3 classes.(a)(b)(c) shows three user inputs corresponding to the three updates. Red circles mark Y_1 and purple circles mark Y_2 in each iteration. I values in 3 updates shown in (a)(b)(c) are 0.2, 2 and 0.5, respectively. (d) shows the end result.

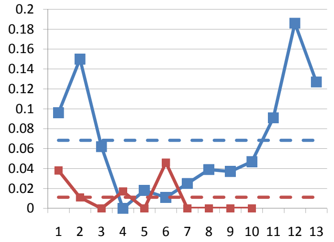

Figure 3: Weights for original and random features: blue squares mark weights for 13 original features, blue dotted line marks the average of 13 weights; read squares mark weights for 10 random features, red dotted line marks the average of these weights.

References

[1] I. Borg and P. Groenen. *Modern Multidimensional Scaling: Theory and Applications*. Springer, 2005.

[2] A. Buja, D. F. Swayne, M. L. Littman, and N. Dean. Interactive data visualization with multidimensional scaling. *Stress The International Journal on the Biology of Stress*, 39(1):1–32, 2004.

[3] A. Endert, C. Han, D. Maiti, L. House, S. Leman, and C. North. Observation-level interaction with statistical models for visual analytics, 2011.

[4] D. H. Jeong, C. Ziemkiewicz, B. Fisher, W. Ribarsky, and R. Chang. ipca: An interactive system for pca-based visual analytics. *Computer Graphics Forum*, 28(3):767–774, 2009.