Automating the Creation of Interactive Glyph-supplemented Scatterplots for Visualizing Algorithm Results

Dinoj Surendran
Department of Computer Science
University of Chicago
dinoj@cs.uchicago.edu

Abstract

Ndaona is a Matlab toolkit to create interactive three-dimensional models of data often found in NLP research, such as exploring the results of classification and dimensionality reduction algorithms. Such models are useful for teaching, presentations and exploratory research (such as showing where a classification algorithm makes mistakes).

Ndaona includes embedding and graphics parameter estimation algorithms, and generates files in the format of Partiview (Levy, 2001), an existing free open-source fast multidimensional data displayer that has traditionally been used in the planetarium community. Partiview[1] supports a number of enhancements to regular scatterplots that allow it to display more than three dimensions' worth of information.

1 Supplemented Scatterplots

Scatterplots are not the most efficient way of representing information (Grinstein et al., 2001). However, they are intuitive and stable (Wong and Bergeron, 1997), and can be supplemented in several ways. We describe some of these augmentations in Section 1, basic Ndaona usage in Section 2, and finally a couple of some embedding methods in Section 3.

[1] http://niri.ncsa.uiuc.edu/partiview

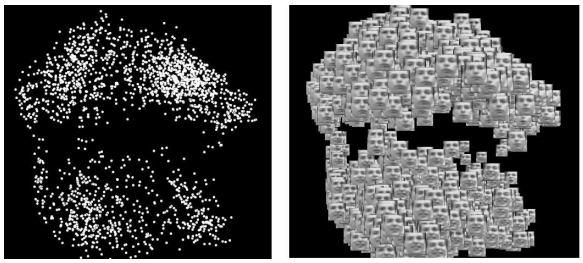

Figure 1: Regular and glyph-supplemented scatterplots showing how a linear kernel can separate happy and sad faces from the Frey Face dataset.

1.1 Glyphs

Glyphs are icons that provide a visual representation of a datum. A simple example of a glyph is a filled-in circle whose color and size convey two additional dimensions of information. More complex examples have been designed to present more information (Tukey and Tukey, 1981) (Ward, 2002). Partiview can use any image as a glyph, as long as all images used can fit in graphics memory.

For example, Figure 1 shows faces from the Frey Face Dataset[2] in linear kernel space; two faces are close together then the vectors u and v representing have a high value of $u^T v$. In this case, each point has a natural visual representation — the face itself. And we therefore use the faces as lossless glyphs, with each glyph representing a 560-dimensional vector (20 x 28 pixel image).

A second example is in Figure 2. It shows Mandarin syllables in a tone recognition experiment (Surendran et al., 2005), with two syllables close

[2] Thanks to Sam Roweis for placing this data on his site.

Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume, pages 280–283,
New York City, June 2006.

Figure 2: A close-up screenshot of a 3D glyph-supplemented scatterplot showing the performance of a linear Support Vector Machine (SVM) on a 4-way Mandarin syllable tone classification task. Ndaona embedded syllables so that those classified similarly by the SVM are close together. The glyph for each syllable represents the 20-dimensional vector input to the SVM. Syllables with the same tone are represented by glyphs of the same color; the white syllables in the foreground have falling tone.

together if the classification algorithm made similar predictions of their tone. The algorithm received for each syllable a 20-dimensional vector that described its normalized pitch contour of the syllable. In this case, a histogram of the pitch contour, with the area between the pitch contour and the horizontal axis shaded for enhanced visibility, results in a highly informative glyph.

A close-up look at the low tone syllables reveals that the algorithm 'thinks' that any syllable whose pitch contour decreases towards the end has a falling tone which is what linguists expect. We can also tell that many of the mistakes made by the algorithm are due to the features and not the algorithm itself. For instance, the several high tone syllables that are close to the cluster of low-tone-syllables (and would thus be classified as having low tone by the algorithm) do in fact have a falling pitch contour.

1.2 Three Dimensions

Partiview users can smoothly spin, zoom, and move the 3d scatterplot model even when it contains hundreds of thousands of points. Interaction and motion deal with difficulties (described in the information visualization and statistical graphics literature) of visually estimating distances when a third dimension is used (Jacoby, 1998).

1.3 Graphs

While Partiview is not the best software for displaying graphs, it can display lines of varying width and color. This permits two bits of information to be displayed about binary relations between points.

1.4 Information about points

While text labels can be placed at points, this often results in visual clutter. It is better to give the user the option of only having labels displayed when actively requested. This option is called 'linking'.

Partiview has some linking capability. When a user clicks on a point (and presses 'p'), the command window displays information about it, such as its position, several features, and any 'comment' provided by the user. For example, Figures 3 and 4 show the results of a 13-class dialog act classification task — the user supplied as comments the words said during each dialog act. Some of these can be seen in the command window of each screenshot.

1.5 Brushing

Brushing is the ability for users to select subsets of data points and apply certain operations to them, such as toggling their visibility (masking), changing their color or size (Becker and Cleveland, 1987).

Partiview supports this very well, and it is possibly the most important feature available for data exploration. For example, we can change the colors of points to be that of any attribute of the data, including its original features. This helps investigate what original features the algorithm is actually using.

For example, in Figure 3 color represents class, while in Figure 4 color represents duration. The color could just as easily be changed to represent other attributes of the data; Ndaona estimates Partiview parameters required for consistent behavior across attributes by normalizing the color map for each attribute.

1.6 Animation

Partiview supports animated data. Ndaona has been written so that one can deal with various combina-

tions of static and dynamic (time-varying) graphical elements, such as fixed points and varying edges, or dynamic points and fixed edges (i.e. the edges always join the same points), or both dynamic points and edges, fixed points and dynamic attributes, and so on. The only difference to the user is that s/he provides a cell array (list) of matrices for the dynamic element instead of a single matrix.

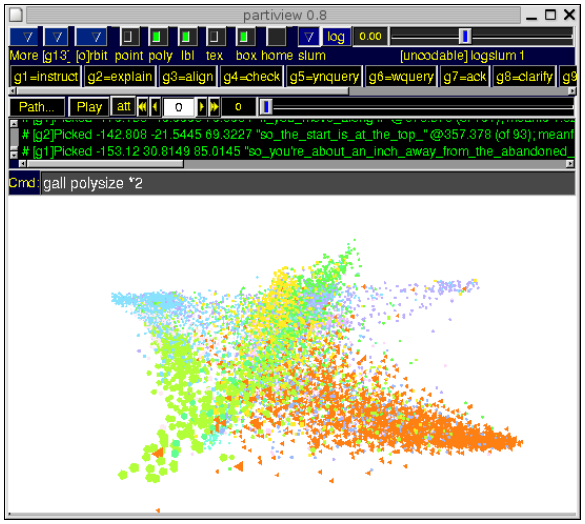

Figure 3: Partiview screenshot of a Ndaona-made model showing the result of a dialog act classification algorithm. Each point represents a dialog act, and all acts of the same type are colored identically.

2 Usage

For flexibility of input, arguments to Ndaona are supplied in parameter-value pairs. For example, say `P` is a $N \times 3$ matrix representing the 3D coordinates of N points and `Q` is a list of N images representing the glyphs of each point. Ndaona includes tools to create such images, or the user can provide their own JPEGs. Either of the equivalent commands

```
ndaona('POSITIONS',P,'PICTURES',Q)
ndaona('POS',P, 'PICS',Q)
```

creates a 3D model with the pictures in `Q` represented at the positions for each point, such as that in Figures 1 and 3. Graphics parameters controlling picture sizes are automatically estimated by Ndaona.

Now suppose that the N points have time-varying positions. Making `P` a list of $N \times 3$ matrices and

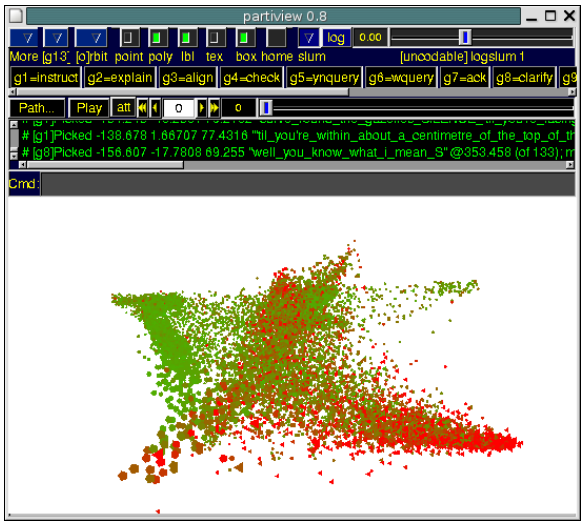

Figure 4: As for Figure 3 but now color represents duration. Shorter acts (top left) are green.

using the same command as above creates a time-varying scatterplot with the images moving about.

If this was classification and the true class of each point is stored in a N-dimensional vector `L`, then

```
ndaona('POS',P,'PICS',Q,'CLASS',L)
```

creates the 3D model with the pictures colored by class, with the same color for points of the same class, such as that in Figure 2. Also, Partiview provides a button for each class that toggles the visibility of data points in that class. If each point has A attributes stored in a $N \times A$ matrix `F`, then

```
ndaona('POS',P,...,'ATTRIBUTES',F)
```

creates a model as before, but with brushing available. The colors of each point can be changed according to the r-th attribute by typing "`color a`r", where `a`r is the automatically assigned name for the r-th attribute. (Users can provide attribute names with another parameter-value pair.)

If the N points also form the nodes of a (connected or not) graph with N_e edges, then if the edges are represented by a $N_e \times 3$ matrix or a sparse $N_e \times N_e$ matrix `G`, the command

```
ndaona('POS',P,...,'EDGES',G)
```

creates the same scatterplot, overlaid with edges.

Additional parameter-value pairs in Ndaona can be used to fine-tune graphics parameters, create files in directory structures that are easy to compress and distribute, change background color, etc.

3 Embedding into Three Dimensions

When visualizing the results of algorithms, users may not have a three-dimensional embedding already available. However, algorithms have been proposed to produce such embeddings, and we now describe some of those available in Ndaona. Ndaona also implements basic dimensionality reduction algorithms such as Principal Components Analysis, Laplacian Eigenmaps, and Isomap.

3.1 Classification Probabilities

If users have a $N \times K$ matrix S of prediction probabilities from a K-class classification algorithm, with $\mathsf{S}(n, k)$ having the probability (estimated by the algorithm) that the n-th point is in class k, then this can be supplied instead.

Ndaona uses the Parametric Embedding algorithm (Iwata et al., 2004) to find a low-dimensional embedding of the N points so that pairs of points that were given similar predictions by the classification algorithm (i.e. have low Kullback-Leibler distance between their prediction probability distributions) are closer together.

3.2 Kernel Matrices

Support vector machines (SVMs) and related methods depend on pairwise similarities of points, in the form of a kernel matrix whos (i, j)-th entry represents the similarity of the i-th and j-th points. Shawe-Taylor and Christianini (2004) suggest using the eigenvectors corresponding to the three smallest positive eigenvalues of the Laplacian of the $N \times N$ kernel matrix to define a $N \times 3$ positions matrix. Ndaona implements an alternative that, in our experience, works better — using the normalized Laplacian of the kernel matrix (with negative entries replaced by zero).

4 Conclusion

Ndaona is an interface package that helps reearchers produce compelling visual representations of their data. Its output is a (time-varying) 3d model that can be displayed by Partiview, an external data viewer. Future plans include adding more scalable embedding algorithms, and allowing other output formats. Ndaona, documentation, and examples of models created with it, can be found at http://people.cs.uchicago.edu/∼dinoj/ndaona

References

R A Becker and W S Cleveland. 1987. Brushing scatterplots. *Technometrics*, 29(2):127–142.

G Grinstein, M Trutschl, and U Cvek. 2001. High-dimensional visualizations. In *Proceedings of the 7th Data Mining Conference-KDD*.

T Iwata, K Saito, N Ueda, S Stromsten, T L Griffiths, and Joshua B Tenenbaum. 2004. Parametric embedding for class visualization. In *Advances in Neural Information Processing Systems 17*.

William G. Jacoby. 1998. *Statistical Graphics for Visualizing Multivariate Data*. Sage University Papers Series on Quantitative Applications in the Social Sciences 07-120, Thousand Oaks, CA.

Stuart Levy. 2001. Interactive 3-d visualization of particle systems with partiview. In *Astrophysical Supercomputing Using Particles (I.A.U. Symposium Proceedings)*, volume 208, pages 85–91. International Astronomical Union.

John Shawe-Taylor and Nello Cristianini. 2004. *Kernel Methods for Pattern Analysis*. Cambridge University Press.

Dinoj Surendran, Gina-Anne Levow, and Yi Xu. 2005. Tone recognition in mandarin using focus. In *Proceedings of the 9th European Conference of Speech Communication and Technology*.

P A Tukey and J W Tukey. 1981. Summarization; smoothing; supplementing views. In Vic Barnett, editor, *Interpreting Multivariate Data*, pages 245–275. John Wiley and Sons.

Matthew O. Ward. 2002. A taxonomy of glyph placement strategies for multidimensional data visualization. *Information Visualization*, 1(3/4):194–210.

Pak Chung Wong and R Daniel Bergeron. 1997. Thirty years of multidimensional multivariate visualization. In Gregory M Nielson, Hans Hagan, and Heinrich Muller, editors, *Scientific Visualization - Overviews, Methodologies and Techniques*, pages 3–33, Los Alamitos, CA. IEEE Computer Society Press.