Dimensionality Reduction for Visualization of Text Similarity

Russell Yanofsky (rey4@columbia.edu)

December 11, 2007

Summary

- Applying Kernel Principal Component Analysis (KPCA), Semidefinite Embedding (SDE) and Minimum Volume Embedding (MVE) to labeled text data to reduce dimensionality, and view relationships between documents in a collection of documents in 2d.
- Potentially useful for research, to visualize output of classification algorithms. Could also be applied on the web to provide visual navigation of search results, related article listings, etc.

Principal Components Analyis (PCA)

- Given cloud of points centered around mean, wide in some directions, narrow in others
- Compute eignenvectors of covariance matrix, $C = \frac{1}{M} \sum_{j=1}^{M} (x_j - \overline{x})(x_j - \overline{x})'$
- Gives set of orthogonal axes through mean point, aligned with directions of greatest variance
- Best 2-D projection (capturing spread of data) is in plane of two axes with highest eigenvalues

Kernel PCA

- PCA can be performed using dot products between points instead of point coordinates (Gram matrix instead of covariance matrix)
- Kernel functions between two points can be substituted for dot products allowing non-linear extension of PCA (for visualization, this means non-linear projections)
- Steps
  1. Compute gram matrix $K_{ij} = k(x_i, x_j)$
  2. Center matrix $\widetilde{K}_{ij} = K_{ij} - \frac{1}{M}\sum_{m=1}^{M} K_{mj} - \frac{1}{M}\sum_{n=1}^{M} K_{in} + \frac{1}{N^2}\sum_{m,n=1}^{M} K_{mn}$
  3. Find eigenvectors and eigenvalues (eigenvectors $\overrightarrow{\alpha_n}$ normalized so that $\lambda_k\,(\overrightarrow{\alpha_k} \cdot \overrightarrow{\alpha_k}) = 1$ for all k)
  4. Take projection of points, $V\Lambda$, where V is eigenvector matrix, Λ is matrix with $\sqrt{\lambda_1} \ldots \sqrt{\lambda_M}$ along diagonal
  5. Plot 2D visualization using 2 coordinates with highest eigenvalues, dropping other coordinates

Semidefinite Embedding

- Builds on Kernel PCA, finding optimal Kernel Matrix using semidefinite programming.
- Given a set of "neighbors" for each point, maintains distances between neighboring points while maximizing distances between unconstrained points.
- Advantage: If points are scattered on high dimensional manifold which twists and curves, fixing neighboring points and spreading out distant points "flattens" manifold, giving good visualization of points lying on it.
- Program: Maximize $tr(K)$ (distance between points), subject to $K \succ 0$ (positive semidefinite kernel matrix), $\sum_{ij} K_{ij} = 0$ (centers kernel matrix), and $K_{ii} + K_{jj} - K_{ij} - K_{ji} = G_{ii} + G_{jj} - G_{ij} - G_{ji}$ where $G_{ij} = x_i \cdot x_j$ for all points i and j which are neighbor of each other or a common point (preserves local distances).

Minimum Volume Embedding

- Builds on Semidefinite embedding. Instead of maximizing $tr\left(K\right) = \sum_{i=1}^{N} \lambda_i$, seeks to maximize $\sum_{i=1}^{d} \lambda_i - \sum_{i=d+1}^{N} \lambda_i$
- Instead of maximizing distance between point in every dimension, only maximizes distance in the first d dimensions being visualized, and minimize distance in remaining dimensions.
- This does a better job "flattening" the data, minimizing volume behind it, at cost of being more expensive to compute.
- Algorithm is iterative, taking existing kernel matrix, K, finding eigenvectors $\vec{v_i}$, performing modified SDE to minimize $tr\left(K\left(-\sum_{i=1}^{d} \vec{v_i}\vec{v_i}' + \sum_{i=d+1}^{N} \vec{v_i}\vec{v_i}'\right)\right)$, and repeating with new K until convergence.

Text Representation

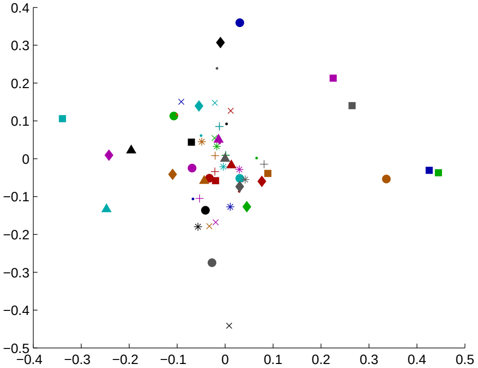
- Each document is represented as a vector of word counts, normalized so documents of differing lengths can be compared.
- *Stemming* algorithm maps related words with different suffixes. For example, "computes", "computing", "computer", mapped into a canonical word stem form, "comput".
- To cut down on irrelevant features, a stop word list is used to remove words like "and", "or", and "the" from feature vectors.
- Remaining words are weighted by *inverse document frequency* (IDF), which is just one over the total number of documents a word appears in.
- RBF kernel used with above preprocessing steps, since this kernel and representation have been shown to be effective for text categorization (Joachims 98)

Experiment

- Experiment was to do two dimensional visualization with issue documents from 2008 presidential candidates' campaign web sites.
- Each document was labeled for the its topic (environment, healthcare, foreign policy, etc.), and the candidate whose views it expressed (Clinton, Giuliani, etc.)
- Having dataset with two distinct labels for each point, makes visualization more interesting, makes it easier to look for patterns in output.

Results

- In progress...
- So far no clear or especially meaningful patterns in visualizations have emerged
- Sample visualization: Kernel PCA with Gaussian kernel, each candidate a different color, each topic a different symbol



References

- T. Joachims. Text categorization with Support Vector Machines: Learning with many relevant features. In Machine Learning: ECML-98, Tenth European Conference on Machine Learning, pp. 137–142, 1998.
- B. Scholkopf, A. Smola, and K. Muller. "Nonlinear component analysis as a kernel eigenvalue problem." *Neural Computation*, 10, 1998.
- B. Shaw and T. Jebara. "Minimum Volume Embedding." Artificial Intelligence and Statistics, AISTATS, March 2007.
- K. Q. Weinberger, F. Sha, and L. K. Saul. "Learning a kernel matrix for nonlinear dimensionality reduction." In Proceedings of the Twenty First International Conference on Machine Learning (ICML-04), pages 839–846, Banff, Canada, 2004.